Born Approximation

Rafael E. Banchs

INTRODUCTION

This report discuss one class of the local search algorithms to be used in the inverse modeling of the time harmonic field electric logging problem (THFEL), the Born approximation. First, a brief analysis about the technique is presented. Then, its equivalence to the least mean square error problem will be shown. Finally, a brief discussion on convergence is presented.

THE BORN APPROXIMATION

As it was presented in [1], the Born approximation is a local search algorithm. Although, strictly speaking, the Born approximation is actually a linear direct inversion technique, it also may be used as a inverse modeling technique by implementing a recursive procedure of successive Born approximations. In this case, it constitutes a local search algorithm because it uses the information in the local derivatives of the objective function in order to search for a minimum.

As it is already known, the Taylor's series expansion for a continuous function f(x) in some neighborhood around x_0 is given by:

$$f(x) = f(x_0) + \left.\frac{df}{dx}\right|_{x_0} (x - x_0) + \frac{1}{2}\left.\frac{d^2 f}{dx^2}\right|_{x_0} (x - x_0)^2 + \frac{1}{6}\left.\frac{d^3 f}{dx^3}\right|_{x_0} (x - x_0)^3 + \ldots \quad (1)$$

where the first order approximation would be defined by the two first terms of the summation:

$$f(x) \approx f(x_0) + \left.\frac{df}{dx}\right|_{x_0} (x - x_0) \quad (2)$$

The Born approximation for an unknown model x, given the inversion data f(x) and the starting model x_0 is defined from (2) as:

$$\hat{x} = x_0 + \frac{f(x) - f(x_0)}{(df/dx)\big|_{x_0}} \tag{3}$$

Notice that what (3) actually represents is nothing more than a model updating equation, where the updated model $\hat{x}$ is computed by adding a "jump" or increment value to the initial model x_0. In the iterative Born approximation or, for simplicity, Born approximation, the resulting updated model at each iteration is used as starting model for the next iteration.

Now, let us rewrite (2) and (3) in a more suitable way for their application to the THFEL problem. In the THFEL problem, f(x) is represented by a set of given measurements and x and x_0 by a set of logarithmic conductivities. Then, f(x) must be replaced by a multivariable vector function $\bar{f}(\bar{x}): R^M \Rightarrow R^N$; where N and M are the dimensionalities of the data space (set of given measurements) and the model space (set of conductivities) respectively. In this way:

$$\bar{m} \approx \bar{f}(\bar{x}) = \bar{f}(\bar{x}_0) + \mathbf{df}\,(\bar{x} - \bar{x}_0) \tag{4}$$

$$\hat{\bar{x}} = \bar{x}_0 + \mathbf{W}\left(\bar{m} - \bar{f}(\bar{x}_0)\right) \tag{5}$$

where $\bar{m}$ is the measurement vector, $\mathbf{df}$ is the MxN matrix of derivatives and $\mathbf{W}$ is the pseudo-inverse of $\mathbf{df}$, which reduces to $\mathbf{df}^{-1}$ when M=N. The entries of $\mathbf{df}$ are given by:

$$df(i,j) = \frac{df_i}{dx_j}\bigg|_{\bar{x}_0} \qquad \text{for } i = 1,2,...N \quad \text{and} \quad j = 1,2,...M \tag{6}$$

and the pseudo-inverse matrix is defined by:

$$\mathbf{W} = \left(\mathbf{df}^T\,\mathbf{df}\right)^{-1}\mathbf{df}^T \tag{7}$$

BORN APPROXIMATION AND THE LEAST MEAN SQUARE ERROR

In this section, it will be shown that the optimization process performed by the Born approximation is totally equivalent to solving the least mean square error (LMSE) problem, in which the mean square error between the inversion data and the model response is to be

minimized. In other words, what the Born approximation actually does is to minimize an objective function that is given by the mean square error function.

First, let us consider the expression in (4). In it, $\bar{f}(\bar{x})$ can be interpreted as an estimate of $\bar{m}$, and according to this it will be rewritten as $\hat{\bar{m}}$. Also, in order to simplify the notation, (4) is going to be rewritten as:

$$\bar{m} \approx \hat{\bar{m}} = \bar{f} + \mathbf{df}\,(\bar{x} - \bar{x}_0) \tag{8}$$

where it is understood that $\bar{f}$ and **df** are evaluated at $\bar{x}_0$. Notice then that, for a given $\bar{x}_0$, $\hat{\bar{m}}$ is a function of $\bar{x}$; but in general it is actually a function of $\bar{x}_0$ and $\bar{x}$.

The mean square error between the given data and its estimate can be then defined as:

$$E^2(\bar{x}) = \frac{1}{N}\sum_{i=1}^{N}\left(m_i - \hat{m}_i\right)^2 \tag{9}$$

Let us now compute the minimum of (9). It can be analytically accomplished by simultaneously equating all its first order derivatives to zero. The derivative of the mean square error with respect to the kth component of $\bar{x}$ is given by:

$$\frac{\partial E^2}{\partial x_k} = \frac{-2}{N}\sum_{i=1}^{N}\left(m_i - \hat{m}_i\right)\frac{\partial \hat{m}_i}{\partial x_k} \tag{10}$$

Or, by substituting (8) into (10):

$$\frac{\partial E^2}{\partial x_k} = \frac{-2}{N}\sum_{i=1}^{N}\left(m_i - f_i - \sum_{j=1}^{M} df(i,j)\,(x_j - x_{0j})\right)\frac{\partial f_i}{\partial x_k} \qquad \text{for } k = 1,2,...M \tag{11}$$

By replacing (6) into (11), rearranging some terms and equating to zero we get:

$$0 = \sum_{i=1}^{N}\left(m_i - f_i\right)\frac{\partial f_i}{\partial x_k} - \sum_{i=1}^{N}\sum_{j=1}^{M}\frac{\partial f_i}{\partial x_k}\frac{\partial f_i}{\partial x_j}\,(x_j - x_{0j}) \qquad \text{for } k = 1,2,...M \tag{12}$$

which defines the system of equations that must be solved in order to obtain the $\bar{x}$ that minimizes the mean square error function (9).

After additional manipulations, (12) can be rewritten in matrix form as:

$$\bar{0} = \mathbf{A}\left(\bar{m} - \bar{f}\right) - \mathbf{B}\left(\bar{x} - \bar{x}_0\right) \tag{13}$$

where $\mathbf{A}$ is an MxN matrix with entries defined by:

$$a(k,i) = \frac{\partial f_i}{\partial x_k} \qquad \text{for } k = 1,2,...M \quad \text{and } i = 1,2,...N \tag{14.a}$$

and $\mathbf{B}$ is an MxM matrix with entries defined by:

$$b(k,j) = \sum_{i=1}^{N} \frac{\partial f_i}{\partial x_k} \frac{\partial f_i}{\partial x_j} \qquad \text{for } k = 1,2,...M \quad \text{and } j = 1,2,...M \tag{14.b}$$

By comparing (6) with (14.a) and (14.b) the following relations can be deduced:

$$\mathbf{A} = \mathbf{df}^T \quad \text{and} \quad \mathbf{B} = \mathbf{A}\,\mathbf{A}^T = \mathbf{df}^T\mathbf{df} \tag{15}$$

Finally, by substituting (15) into (13) and solving for $\bar{x}$:

$$\bar{0} = \mathbf{df}^T\left(\bar{m} - \bar{f}\right) - \mathbf{df}^T\mathbf{df}\left(\bar{x} - \bar{x}_0\right) \quad \Rightarrow \quad \bar{x} = \bar{x}_0 + \left(\mathbf{df}^T\mathbf{df}\right)^{-1}\mathbf{df}^T\left(\bar{m} - \bar{f}\right) \tag{16}$$

which is exactly the Born approximation as it was defined in (5).

In summary, the Born approximation method can be interpreted as a gradient method [2] that uses an exact line search algorithm and an objective function given by the mean square error between the inversion data and the synthetic data, which is computed by using a first order expansion of the forward modeling function around the current model. Notice that the way in which the synthetic data is computed determines one of the most important peculiarities of the Born approximation, that is the fact that the objective function changes at every iteration. As it can be seen from (8) and (9), the error is actually a function of $\bar{x}$ and the current model $\bar{x}_0$.

A DISCUSSION ABOUT CONVERGENCE

Because of the particular way in which the Born approximation moves through the model space, convergence assumptions are not as obvious as in the case of other local search techniques. In this section, a very simple case is going to be studied. It is expected from this analysis to develop a more intuitive idea of how the Born approximation works and to provide a better understanding of its convergence properties.

Let us consider a very simple problem in which the inversion data set is conformed of two values, m_1 and m_2; and the inversion model consists of just one unknown. According to the Born approximation methodology, given an initial model x_0, the new estimate will be given by:

$$\hat{x} = x_0 + \Delta(x_0) \tag{17}$$

where $\Delta(x)$ is going to be referred as the increment function, and it is given by:

$$\Delta(x) = \frac{(m_1 - f_1)df_1 + (m_2 - f_2)df_2}{df_1^2 + df_2^2} \tag{18}$$

where f_1 and f_2 are the components of a function $\bar{f}$, such that $\bar{f}(x_s) = [f_1(x_s)\ \ f_2(x_s)]^T = [m_1\ \ m_2]^T$ if x_s is the correct solution model.

Notice that, for practical reasons, a simplified notation has been introduced in (18) and will be used all through this analysis. The complete notation equivalents are given by:

$$f_i = f_i(x); \quad df_i = \frac{df_i}{dx}; \quad df_i^2 = \left(\frac{df_i}{dx}\right)^2 \quad \text{and} \quad d^2 f_i = \frac{d^2 f_i}{dx^2} \tag{19}$$

From (17) it can be observed that the possible points of convergence are given by those models that make the increment function equal to zero. Then, by equating (18) to zero, a convergence condition over the model space is defined:

$$(m_1 - f_1)df_1 + (m_2 - f_2)df_2 = 0 \tag{20}$$

Let us now consider a possible point of convergence x_c. (Notice that, although it constitutes a solution of the inverse problem, it does not necessarily have to be equal to the correct solution x_s.) Suppose that the initial model for the iteration under consideration x_0 is very close to that

possible point of convergence x_c; so close that the behavior of the increment function in that neighborhood can be appropriately approximated by a line. Depending on the slope of such a line, four different situations can arise.

A.- Non-oscillatory Divergence.

This situation, which is illustrated in Figure 1.a, occurs when the first derivative of the increment function at x_c happens to be greater than zero. As it can be seen from Figure (1.a), it does not matter if the value of the initial model x_0 is less or greater than x_c, the sign of the resulting increment function it such that x_0 will be always pushed away from x_c.

B.- Non-oscillatory Convergence.

This situation, which is presented in Figure 1.b, arises when the first derivative of the increment function at x_c is in the interval (-1, 0). Notice that, in this case, the increment function pushes x_0 towards x_c. As the magnitude of the increment is smaller that the distance between x_0 and x_c, convergence towards x_c will be achieved after successive iterations.

C.- Oscillatory Convergence.

This situation occurs when the first derivative of the increment function at x_c is inside the interval (-2, -1). As it can be seen from Figure 1.c, in this case convergence is also achieved; but it is done in an oscillatory fashion. This is because the size of the increment is greater than the distance between x_0 and x_c. In this way, in each successive iteration, the updated value of x will be in the opposite side of x_c.

D.- Oscillatory Divergence.

In this last situation, which arises when the first derivative of the increment function at x_c happens to be less than -2, divergence occurs in an oscillatory fashion. This is, as it can be seen from Figure 1.d, because the size of the increment will be always greater than $2|x_c - x_0|$.

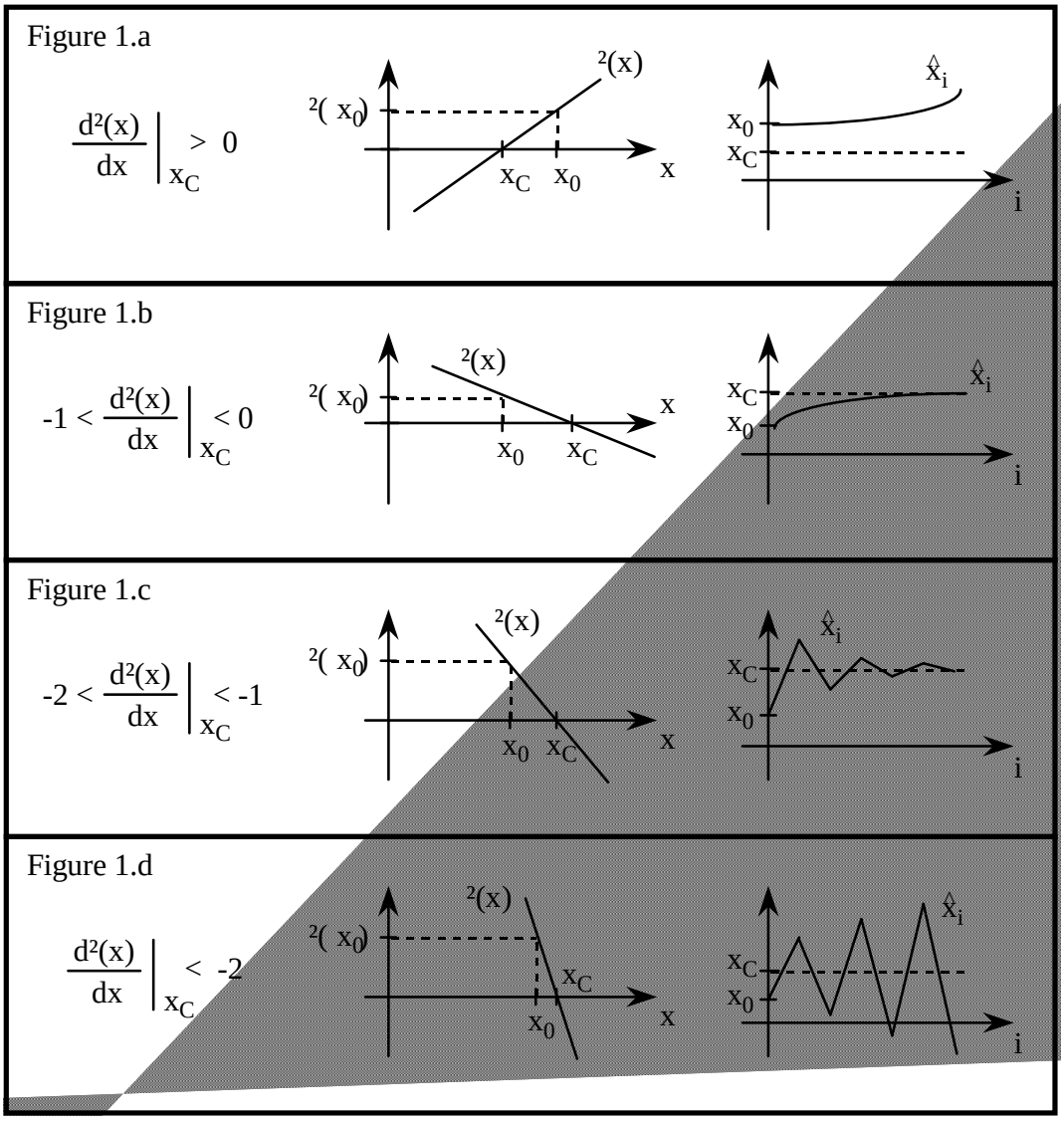

Figure 1: Possible situations in the vicinity of a point of convergence x_c.

Notice also that very particular kinds of behavior occur when the first derivative of the increment function at x_c takes the specific values of 0, -1 or -2. In the first case, the size of the increment will be zero and so the algorithm will get stuck in x_0; in the second case, the algorithm will reach x_c in just one iteration; and in the last case, the algorithm will keep oscillating forever between x_0 and $(2x_c - x_0)$. However, these situations are very unlikely to occur in the practice because of the following two reasons. First, in general, the behavior of the increment function around x_c is

not actually linear; and second, after few iterations, the roundoff errors due to numerical computation will eventually deviate the algorithm from such kind of punctual conditions.

Based on the observation described before, it is possible to define a stability condition. Given a convergence point x_c, its is going to be defined as a stable convergence point if only if the situations B or C discussed above occurs in its vicinity. That is, if the first derivative of the increment function at x_c is in the interval (-2, 0).

$$-2 < \left.\frac{d\Delta(x)}{dx}\right|_{x_c} < 0 \tag{21}$$

where, by differentiating (18):

$$\frac{d\Delta(x)}{dx} = -1 + \frac{(m_1 - f_1)d^2f_1 + (m_2 - f_2)d^2f_2}{df_1^{\,2} + df_2^{\,2}} - 2\frac{\left[(m_1 - f_1)df_1 + (m_2 - f_2)df_2\right]\left[df_1\, d^2f_1 + df_2\, d^2f_2\right]}{\left[df_1^{\,2} + df_2^{\,2}\right]^2} \tag{22}$$

Notice that when (22) is evaluated at a possible convergence point, according to (20) the last term reduces to zero. In this way:

$$\frac{d\Delta(x)}{dx} = \frac{(m_1 - f_1)d^2f_1 + (m_2 - f_2)d^2f_2}{df_1^{\,2} + df_2^{\,2}} - 1 \qquad \text{when } x = x_c \tag{23}$$

Now let, us consider the equivalent LMSE problem to the one under analysis. By considering $x_0 = x$, the corresponding error function and its first and second order derivatives are given by:

$$E = \frac{1}{2}(m_1 - f_1)^2 + \frac{1}{2}(m_2 - f_2)^2 \tag{24}$$

$$\frac{dE}{dx} = -(m_1 - f_1)df_1 - (m_2 - f_2)df_2 \tag{25}$$

$$\frac{d^2E}{dx^2} = (df_1^{\,2} + df_2^{\,2}) - \left[(m_1 - f_1)d^2f_1 + (m_2 - f_2)d^2f_2\right] \tag{26}$$

By equating the first derivative to zero the next expression follows:

$$(m_1 - f_1)df_1 + (m_2 - f_2)df_2 = 0 \tag{27}$$

which is exactly the same convergence condition defined in (20). What this means is that the possible points of convergence x_c for the Born approximation algorithm are precisely the local extremes and saddle points of the mean square error function.

Let us consider the particular case of the local maxima by imposing a negative concavity condition on the second order derivative. By doing so:

$$(df_1^2 + df_2^2) - [(m_1 - f_1)d^2f_1 + (m_2 - f_2)d^2f_2] < 0$$

$$\Rightarrow\ 0 < \frac{(m_1 - f_1)d^2f_1 + (m_2 - f_2)d^2f_2}{df_1^2 + df_2^2} - 1 \tag{28}$$

where the right hand side of the inequality is the derivative of the increment functions as it was defined in (23). So, (27) represents the condition under which situation A occurs (see Figure 1.a). This means that local maxima are unstable convergence points.

On the other hand, by doing a similar analysis, local minima satisfy the following condition:

$$0 > \frac{(m_1 - f_1)d^2f_1 + (m_2 - f_2)d^2f_2}{df_1^2 + df_2^2} - 1 = \left.\frac{d\Delta(x)}{dx}\right|_{x_c} \tag{29}$$

which includes situations B, C and D (see Figure 1). Notice that according to this, every stable convergence point will always correspond to a minimum (or a saddle point) of the mean square error function along $x_0 = x$, but not necessarily every minimum will correspond to an stable convergence point.

A stability condition for minima can be derived from (21) and (29). It is given by:

$$df_1^2 + df_2^2 + (m_1 - f_1)d^2f_1 + (m_2 - f_2)d^2f_2 > 0 \tag{30}$$

Notice that for the correct solution x_s (30) will always hold since $(m_1 - f_1) = (m_2 - f_2) = 0$. This means that the correct solution will always be a stable convergence point.

The particular case of saddle points occur when the inequality in (29) is replaced by an equality. This correspond to a very specific situation in which convergence may or may not be achieved depending on the peculiarities of the problem. However, it can be intuitively seen that in the concave neighborhood of a saddle point there exists a high chance for stability.

VARIATIONS OF THE BORN APPROXIMATION

The pure Born approximation as it has been defined above can present some practical problems when implemented. For example, when the derivatives of the function are relatively much more smaller than the differences between it and the inversion data, huge "jumps" in the model space can result from the model updating. This can make convergence very difficult in the case of non-linear functions; or, what is worse, can send the model out of the domain of the problem. For this and some other reasons, additional parameters have been introduced into the Born approximation algorithm in order to improve its performance.

1.- Relaxation factor.

The introduction of a relaxation factor that multiplies the increment size, also known as successful over relaxation (SOR), helps to avoid large jumps into the model space. It is actually an over relaxation procedure because it multiplies the increment by some value smaller than 1.0. The relaxation factor can be computed according to different criteria. It can be fixed or it can be varied from an initial value in such a way that it approaches 1.0 as the number of iterations increase. It can be also relative to the size of the increment, so that small increments will be barely altered and large increments will be substantially reduced.

2.- Barrier factor.

Following the same idea of the barrier functions used in gradient methods, a barrier factor can be defined in order to prevent the updated model from getting out of the domain of the problem. Although 'limiting factor' would be a more appropriate designation, the name of barrier factor

will be used because of its analogy to gradient methods' barriers. The barrier factor multiplies the increment and its value is chosen in order to confine the jump inside the allowed region in the model space. In the program implementation, the barrier factor is defined by the following function:

$$bf_i = \left[1 + \exp\left(k\left(\left|x_i + \Delta_i\right| - x_{max}\right)\right)\right]^{-1} \qquad \text{for } i = 1,2,...M \tag{31}$$

where x_i is the ith component of the model, Δ_i is the ith component of the increment, x_{max} defines the domain of the problem ($-x_{max} < x_i < x_{max}$ for i=1,2,...M) and k is a variable parameter that determines the sharpness of the function around the limit value x_{max}. Notice that for k->∞, (31) defines a binary factor that is 0 when the jump leads out of the domain of the problem and is 1 when it remains inside. Notice also that, according to (31), a different barrier factor is defined for each of the components of the model.

3.- Nomalization of equations.

With this feature, each of the individual equations in (4) is divided by its associated measurement. This is done in order to balance the contributions of all the different elements in the inversion data set. It is totally equivalent to the normalization of the error terms used in gradient methods [2]. By using normalization, the equations in (4) are rewritten as:

$$1 \approx \frac{f_i}{m_i} + \frac{1}{m_i}\sum_{k=1}^{M} df(i,k)\,(x_k - x_{0k}) \qquad \text{for } i = 1,2,...N \tag{32}$$

CONCLUSIONS

The Born approximation provides a suitable methodology for the inverse modeling of the time harmonic field electric logging problem. This is because it constitute a relatively simple procedure that only relies on first order derivatives, which can be analytically approximated [3].

Although in fact the Born approximation solves a linearized version of the problem at each iteration, as it has been discussed in this report, the possible convergence points of the algorithm are generally determined by the minima of a mean square error function.

REFERENCES

[1] Update Report #12: The Inverse Problem.

[2] Update Report #13: Gradient Methods.

[3] Update Report #8: Recursive computation of the derivatives.